The Boston Computer Society

SINCLAIR-TIMEX USER GROUP NEWSLETTER

Volume 2, Issue 3 March 1983

This newsletter is produced to inform group members of the agenda and logistics for future meetings, as well as to recap and amplify the information provided at the last meeting. It also provides a forum for members and interested parties to communicate what they have learned or developed relating to Sinclair and Timex computer products.

USER GROUP MEETINGS

Date: Wednesday, March 16, 1983
Time: 7:00 p.m.
Place: Large Science Auditorium, UMass, Harbor Campus
(Directions on last page)

At this meeting, Larry Becker will be demonstrating a practical application of the Sinclair-Timex computer. The computer is used as an "intelligent laboratory station terminal." Specifically, the computer collects analog data at a school laboratory work station, and forwards the data to a centralize computer. The complete system can contain up to 16 Sinclair-Timex computers and a central Apple computer. Larry, who is on a sabbatical leave from Siram College in Northeast Ohio, is developing the system at the Technical Education Research Centers (TERC) in Cambridge.

Also planned for the March meeting is a review of the E-Z Key keyboard by Bob Masters and a review of MCODER, an integer compiler, by Will Stackman.

Following the presentation, we want to break up into groups to discuss topics of special interest. At the past few meetings, we have had little time for such groups -- a situation we are trying to correct. Tentatively, an advanced and beginner's group are planned, but groups to discuss the presentations or to discuss FORTH can be formed, depending upon interest of those attending the meeting.

The April meeting will be held on the 13th, the second Wednesday of the month. If you have items to discuss at the meeting or suggestions for presentations, contact Sue or Cliff.

NOTE THE CHANGE OF THE APRIL MEETING DATE.

HIGHLIGHTS OF THE FEBRUARY MEETING

Sue Mahoney opened the meeting with a report of a visit to Sub Oak, New York, in Westchester County. The public library in that community has an interesting new project -- they are lending TS-1000 computers. You can borrow the computer for one week with a library card. There are fines for overdue computers and there's a long waiting list.

HIGHLIGHTS -- Concluded

Mike Coughlin lead a discussion on the use of FORTH with the Sinclair-Timex computers. Mike explained that FORTH is a computer language, as is BASIC, only FORTH's not as easy to learn. After loading the FORTH program, you can write programs in the FORTH language. Mike described the concepts of the FORTH language, such as words, the dictionary, reverse-Polish notation, and threaded interpretation. In FORTH, you generate words instead of statements. Instead of being stored with line numbers in sequence, words are stored in a dictionary. Reverse-Polish notation, similar to that used in some calculators, manulipates a stack, which is used to store numbers like variables do in BASIC. When a program is run, words from the dictionary are interpreted following the logical thread defined by the programmer.

Mike said that the language is likely to be used more in the future because of its power and speed. FORTH is useful in control and interfacing applications. FORTH programs are reported to run as fast in the Slow mode as BASIC programs do in Fast mode. Mike has ZX FORTH from Gladstone Electronics (reviewed below). During the discussions, Bob Smith stated that another version of ZX FORTH was available from The FORTH Dimension, a company in Pennsylvania. There are differences between the two versions, such as in the use of the Sinclair, versus ASCII, character set.

Following Mike's presentation, John Kemeny demonstrated the video monitor interface he purchased from Random Access, Box 4177, Phoenix, Arizona 85080. He previously reported that his computer didn't work after he installed the interface. His problem proved to be a simple soldering error. He correctly soldered the resistors, diodes, transistors, and other delicate electronic components; however, in the output connector, he shorted the two connectors of the coaxial cable. This was quickly fixed once the error was found. He is now very happy with the interface. The interface costs $20.45 postpaid and provides both a television and a video monitor interface, with switchable normal and inverted (white on black) display. John also discussed why one would want to use a video monitor and the criteria he used in selecting his monitor (see page 7).

Peter Nichols gave a demonstration of the Byte-Back modem. With a modem, it is possible to use the computer and a telephone to communicate with bulletin boards and information services, such as the Source. The Byte-Back modem runs at 300 baud and comes with software to allow the Sinclair-Timex to emulate a "dumb" terminal.

REVIEW OF ZX FORTH FROM GLADSTONE by Michael Coughlin

A complete new programming language has become available for the ZX-81 and TS-1000 computers with 16 K memory. Called ZX FORTH, it promises to become an increasingly important language since it allows the very small computers to cope with complicated applications. Originally developed in England by Artic Computing Ltd., and sold through Gladstone Electronics in Buffalo, New York, ZX FORTH is an adaption of FigFORTH. FigFORTH has a good selection of instruction manuals and system expansions supporting it. Unfortunately, this first edition of the language for our computer suffers from a small manual with many errors. Nevertheless, the language is so powerful that experienced software hackers should buy ZX FORTH at once. Beginners are advised to wait until better instructional materials are provided and any bugs in the code have been exterminated. The price for ZXFORTH is $29.95 plus shipping.

REVIEW OF THE ORGANIZER FROM TIMEX by Jack Hodgson

"The Organizer", from Timex, is a data base management program. It helps the user create, maintain, and work with a small data file. It may be easier to think of this program as a box of blank file cards onto which you can write all sorts of information. The Organizer helps you put stuff on these cards, alter and update them, sort the cards, and search through them for certain items.

When you first run the program, you are asked to set up the titles for the field. These are the labels that will appear on all the cards in your file. Next you must tell the program where each "field" is to begin. The fields are the areas into which you will enter actual information of the file. Finally, the program shows you a card (actually known as a record) that has on it the titles you entered earlier, onto which you can now type your data.

After you've entered a few (or more) records you can begin to work with the file. Left to its own devices the program will put the file in order by the first field on the screen, but this can be changed easily so that they are ordered by any field you choose. This is done with the command ORDER.

You move through the file by using the commands FORWARD, BACK, and LIST. The first two will step you through the file one record at a time, and the latter causes the records to appear one at a time for a few seconds each until you stop the listing by pressing a key. RESET will take you from whereever you are in the file back to the first record in one step.

ALTER, DELETE, and ENTER allow you to alter a record, completely delte a record, or add new ones, respectively. SELECT is the most powerful and handiest command in the repertoire. With it you can have the program look through all the records for any words, word, or even a few letters that may exist in the file. The program will display, one at a time, all the records in which the string in question appears. Because I don't own a printer I'm not real familiar with the commands that access it. But evidently PRINT and COPY allow you to print a copy of all the records in the file or of any individual record. Finally INFORM tells you how many records you currently have and how much space is left, and QUIT takes you back to the main menu.

Of all the TS produced software I've seen this one is the handiest (Vu-Calc a close second). The documentation is relatively adequate but, as usual, printed in ludicrously small type. The on-screen prompts are good in that all the directions are there but bad in that you must read them carefully. Their syntax is stilted (to my eye) and they are inconsistant in the way each command is manipulated (sometimes you move the cursor around with the arrows and other times with the shift key, sometimes you exit the command with the enter key and other times with stop). But the instructions are all there on the screen, you just have to be sure to read them carefully.

My biggest problem with this program is its small capacity (a little over 9000 characters for data). Of course this is really limited by the capacity of the machine and the speed of the load routine (It takes 7 plus minutes to load the filled to capacity file) not the program itself. For most home applications that I can think of it's perfectly adequate.

Also included on the cassette is a copy of the organizer filled with information from a world atlas. "The Gazeteer", while of somewhat dubious value as a resource, is a good example of what can be done with the program.

FRESH SQUEEZED COMPUTER DATA

Since the Sinclair-Timex, and many other computers, store characters in 8-bit bytes, it is commonly assumed that 8 bits are required to store a character of text. This is not so. There are methods, called data compression techniques, which allow you to store data in less storage space. The penalty is that it takes time to encode and decode your data.

The amount of data compression that can be achieved depends on the amount of a priori information known about the data. For example, in English text, the letter E occurs far more frequently that any other letter. Therefore, you could save space by using variable length codes with short codes for the letter E and other frequently used characters. In this article we will examine a method of data compression which does not make assumptions about the frequency of use of characters.

Information theory, a field pioneered by Claude E. Shannon in 1949, tells us that if we have r characters to choose from, then at least $\log_2 r$ bits are required to store a character. From the Sinclair-Timex user's manuals, we see that there are approximately 256 different characters (a few are not assigned), with CODEs from 0 to 255. Thus, $\log_2 256 = 8$ bits are required to store an arbritary character.

Does this contradict with what we said earlier? Not if you note that most data uses only a small fraction of those 256 characters. Let's consider text which uses: the digits 0 through 9; the three punctuation points -- period, comma, and question mark; the letters A through Z; and the space character. The minimum storage for text containing just these 40 characters is $\log_2 40$ or approximately 5.3 bits. If we were able to pack the characters into their minimum required storage space, we could save almost 3 bits per character. But we have two problems. Firstly, we can't store data in a fraction of a bit. And, if we use 6 bits for each character, we wastes about 13 percent of the storage space (what a different a bit makes). The second problem is that it is very inconvenient to manipulate 6 bits of data on our computer, even in machine language. The solution lies in bunching characters together. Three characters require 3*5.3 or approximately 15.9 bits. Using 16 bits for three characters doesn't waste much space.

Following is a very flexible, and therefore fairly complex, program to do data compression. ~~You may have difficutly figuring out how it works, thus~~ it is probably best to treat it as a black box -- use it as is.

- M$ is the character set, which in our example is the 40 characters cited above. You could replace these characters with those characters you need in your application, e.g., just the 10 decimal numbers.

- B is the number of bytes that receive one bunch of characters. Here we are packing 3 characters into two bytes, so B is 2. The program figures out N, the number of characters to pack into the B bytes, in this case three. You can find the percent efficiency of your packing by PRINTing 100*(N*LN M)/(8*B*LN 2), where M is LEN M$.

- I$ is the input characters (your data).

- The subroutine at line 200 compresses I$ and returns the result in A$. If the number of characters in I$ is not a multiple of N, the subroutine will pad I$ with the necessary extra character(s).

- The subroutine at line 800 decompresses I$ and puts the result in A$.

Each half of the program will run in 1 K RAM, if the REMs are deleted. (This may not be too useful, however, since there is little room left for data.) To run in a 1 K system:

1. Write two programs -- one with the compress and one with the decompress subroutines.

2. Figure out what your data encodes into using compress.

3. Add this data string and the decompress subroutine to your final program.

If you are familiar with the PDP-11 computers, you probably have seen this compression technique under the name RADIX-50. Why 50? In octal (base 8), 50 is decimal 40.

```
  1 REM  SQUEEZE
  2 LET B=2
  3 LET M$="1234567890 ABCDEFGH
IJKLMNOPQRSTUVWXYZ,.?"
 10 DIM K(B)
 20 LET T=256
 30 LET M=LEN M$
 40 LET N=INT (8*B*LN 2/LN M)
100 INPUT I$
110 GOSUB 200
120 FOR I=1 TO LEN A$
130 PRINT I;"=";A$(I);CODE A$(I
),
140 NEXT I
150 STOP

200 REM
210 LET A$=""
220 IF LEN I$=N*INT (LEN I$/N)
THEN GOTO 250
230 LET I$=I$+M$(M)
240 GOTO 220
250 LET I=0
260 IF I<>N*INT(I/N) THEN GOTO
300
270 FOR K=B TO 1 STEP -1
275 IF I>0 THEN LET A$=A$+CHR$
K(K)
280 LET K(K)=0
290 NEXT K
300 LET I=I+1
310 IF I>LEN I$ THEN RETURN
320 LET J=-1
330 LET J=J+1
340 IF J+1<M AND I$(I)<>M$(J+1)
THEN GOTO 330
350 FOR K=1 TO B
360 LET K(K)=M*K(K)+J
370 LET J=INT (K(K)/T)
380 LET K(K)=K(K)-T*J
390 NEXT K
400 GOTO 260

800 REM DECOMPRESS
810 LET A$=""
820 LET I=0
830 LET I=I+B
840 IF I>LEN I$ THEN RETURN
850 FOR K=1 TO B
860 LET K(K)=CODE I$(I+1-K)
870 NEXT K
880 FOR J=1 TO N
890 LET R=0
900 FOR K=B TO 1 STEP -1
910 LET R=K(K)+T*R
920 LET K(K)=INT (R/M)
930 LET R=R-M*K(K)
940 NEXT K
950 LET A$=M$(R+1)+A$
960 NEXT J
970 LET A$=A$(N+1 TO )+A$( TO N
)
980 GOTO 830
```

RAM-PACK WOBBLE

RAM-pack wobble is a phrase that has been used to describe a problem frequently encountered with the Sinclair-Timex computer with a 16-K RAM pack. (We borrowed the phrase from Tim Hartnell, the British author and publisher who spoke at our December meeting. Tim uses the phrase as the title of a column he writes in a British magazine.) The problem is usually observed when entering information into the machine and results in a system crash. It occurs because of the mechanical design of the RAM pack (reportedly the design is that of Uncle Clive's brother). The RAM pack is cantilevered off the back of the computer such that motion of the RAM pack relative to the computer occurs when keys are pressed. The motion causes a break in electrical contact on the edge connector between the two units.

There are many, many different solutions to the RAM-pack wobble problem. If you have had the computer for a while, you have undoubtedly found a solution. For new users, here is a list of some of the solutions which have been proposed. They are in no particular order, and we don't guarantee any of them.

- Tim Hartnell's solution, as we reported in the January newsletter, is to mount the computer and RAM pack to a shelf with big C-clamps and to use an external keyboard. There are a number of alternatives associated with the use of an external keyboard as to where the computer is placed (e.g., on the shelf, in the base of the keyboard). These solutions work because they eliminate the need to touch the computer.

- Will Stackman, in the missive he handed out at the last meeting, proposed Blue Goo as "a cheap and final solution" to the problem. He says to buy a package of Fun-Tak synthetic rubber kneadable adhesive at your local five and dime. "Wedge a wad between the RAM pack and the case." Various other solutions using household and silly puddy have been proposed, but this solution is perhaps better as there is little oil is the Blue Goo.

- Similar to the Blue Goo solution, the use of Velcro has been proposed by several different people. Simply glue a strip to both the computer and the RAM pack and the wobble is gone. Memotech's memory modules come with patches of self-adhesive Velcro.

- One of the frequently cited and simpliest solutions is to pull the RAM pack out a small amount. This solution doesn't require items you may not have in the house. Some have suggested the reverse, pushing the RAM pack in as far as it will go; however, pulling out a silly millimeter or two is probably the more reliable solution.

- More than one person has soldered the RAM pack to the edge connector. Less drastic, but equally effective, ribbon cable can be soldered to both units providing a permament connection. It has also been suggested that the edge connector be replaced with a more reliable connector.

- Various arrangements of elastics have proved effective. Gene Blumenreich says to wrap three large rubber bands, of the type found on the Boston Sunday Globe, around the base of the computer and RAM pack. Be sure to place the rubber bands so they don't go over the keyboard. Place a book or magazine under the computer, but not the RAM pack. Don't let anything hit the RAM pack. Gene writes that "the resulting structure looks messy, but it is pretty stable."

MORE RAM-PACK WOBBLE SOLUTIONS

- One company is selling a flexible cable interface between the computer and RAM pack. Another company manufactures a plastic guard which surrounds the RAM pack to prevent bumping of the unit.

- Wedging erasers and pencils between the computer and the RAM pack have been proposed. The use of contact cleaner and television tuner lubricant have apparently been used by some people to good effect. It has been suggested that the use of a soft surface for the computer, such as a magazine instead of a table, will prevent wobble.

- Finally, Your editor has a 4-inch printed circuit board, with appropriate edge connectors on each end, to separate the computer and RAM pack. In addition to solving the wobble problem, this allows access to the edge connector signals required by other peripherals.

YOU'VE GO THE LOOK

One of the terrific things about using a Sinclair-Timex computer is that it is so educational. And you can learn about more than just BASIC. For example, recently a friend of mine, John Rommelfanger, demonstrated a MicroAce (an obsolete ZX-80 copy) connected to a video monitor. Instantly I knew I wanted a monitor. The reason is very simple -- I liked the look. Although I've read about using a UHF modulator, and other ways to clean up the television picture, there is really no beter way to eliminate eye strain than a monitor.

With the help of John and others, I quickly learned all about what to look for when selecting a monitor. Here's the poop.

First, decide if you will use the monitor just with the Sinclair-Timex, or with other, future, computers (the TS-1000000, perhaps). The most important feature of any monitor is its bandwidth, which is measured in megahertz (MHz). It determines the maximum resolution (or fineness) of the picture. A television has a maximum of 4 1/2 MHz bandwidth, while inexpensive monitors range from 8-22 MHz. Price is almost directly proportional to the bandwidth -- cheaper monitors have less. Because the Sinclair-Timex produces a low resolution picture, it will look virtually the same on any monitor.

Second, check the phosphor. They come not just in black and white, but also in green and amber. Europeans use amber almost exclusively. It is very nice, but there is a slight premium on it in this country. Also, the phosphor speed is important — get P31, a fast phosphor (don't let the salesman convince you that slow phosphors will reduce flicker).

Third, check the input jacks. Most monitors have 75 ohms impedance input. Some also have a high impedance. If you plan to directly connect the wire going into the computers modulator (the little silver box inside) to the jack, you should use the high impedance or build a one transitor emitter-follower circuit. Some people have had success without either, however. In any case, I remmend you build a video inverter circuit, because light on dark looks far better on a monitor. Some monitors come with a nylon antiglare screen. These reduce the brightness slightly, but enhance the contrast. These screens can be purchased separately, but they tend to be expensive.

Finally, check to see if the monitor has a handle. This is often overlooked, but much appreciated when the monitor has to be moved.

GENERAL INFORMATION

Meetings are open to the public; however, attendees are encouraged to join the Boston Computer Society (BCS). This newsletter is free to members. Backissues are one dollar each.

FOR MORE INFORMATION

Sue Mahoney, Director of the Sinclair-Timex User Group
c/o The Boston Computer Society or call (203) 573-5816.

Cliff Danielson, Newsletter Editor
14 Davis Road, Chelmsford, MA 01824, (617) 256-4638.

John Kemeny, Contributing Editor & Correspondent With Other User Groups
284 Great Road, Apt. D5, Acton, MA 01720.

Library Committee: Beth Elloitt, Sean O'Rahilly, and Bob Sanchez.

DIRECTIONS TO THE MEETING

The Sinclair-Timex User Group meets in the Large Science Auditorium (Room 8/2/009) of the University of Massachussets of Boston, Harbor Campus. The Harbor Campus is only 3 miles from downtown Boston and easily accessible by public and private transportation. From the north or west, take the Southeast Expressway to Exit 17. Turn left onto Columbia Road. Enter the rotary and take the first right (Morrissey Boulevard). Bear right on the traffic island, following UMass/Boston sign. Turn left into the Campus. From the south, take Morrissey Boulevard northward to the campus. On the MBTA, take the Red Line (Ashmont Train) to Columbia Station. Transfer to the free University shuttlebus in the T parking lot.

IMPORTANT NOTICE ! If the mailing label on this newsletter is handwritten (except if your a user group), then you are not on the mailing list of the Sinclair-Timex User Group. You need to either join the BCS or, if you are a BCS member, contact Mary McCann in the BCS office to be added to the Sinclair-Timex mailing list.

The Boston Computer Society

Three Center Plaza
Boston, MA 02108
617-367-8080